MUSIC IV PROGRAMMER'S MANUAL

by

M. V. Mathews and Joan E. Miller
Bell Telephone Laboratories, Incorporated
Murray Hill, New Jersey

Contents

Acknowledgement

The authors would like to express their appreciation to Mrs. C. C. Lochbaum for her work in separating BE FAP from BE SYS 5 for use with the Music IV program and the write-up of Section 4.

Section 1. Introduction.

This material is intended to be a handbook and guide to the use of the Music IV program, which is the most recent version of the computer music programs developed at Bell Telephone Laboratories. The material includes descriptive parts intended to introduce the reader to the music routines, cookbook procedures for using the program, and detailed technical descriptions of the programs. The technical descriptions are intended to be sufficiently complete so that a competent programmer can adapt the music for programs for use at any IBM 7090-7094 installation and can modify the programs to meet his own particular musical objectives. This handbook is intended to be accompanied by a tape containing card images of all the music programs and another memorandum describing the operation of the BE FAP compiler. These materials should be sufficient to introduce the Music IV program at another computer installation.

We suggest that the potential user familiarize himself in a general way with the material in this handbook before making a detailed study of the technical parts. The various sections contain the following materials:

Section 2 - This section is an over-all description of the operation of the music program, which might be given in a technical paper. It does not go into sufficient detail to allow the use of the program but provides a good introduction.

Section 3 - This section gives a description of the tape containing the computer music programs and outlines how to get started in adapting these programs to a computer installation.

Section 4 - This section gives the details of the BE FAP compiler. This program is essential for compiling orchestras, and adapting it for use at another computer installation is probably the most difficult programming job required by the music programs.

Section 5 - This section is a cookbook outline of how to execute a music program including compiling an orchestra and playing a score with the orchestra.

Section 6 - This section contains detailed technical descriptions of the various music programs.

Section 2. Generation of music by the Music IV program.

The objective of the Music IV program is to produce a sequence of numbers which can be converted to sound by means of a digital-to-analog converter and a smoothing filter. This process is schematized in Figure 1. It has been described in detail in other publications ("The Digital Computer as a Musical Instrument" by M. V. Mathews, Science, Vol. 142, No. 3592, pp.553-557, Nov. 1, 1963) and little more will be said about it here. The digital-to-analog converter, the smoothing filter and the loudspeaker by which the numerical samples are converted to sound are pieces of physical equipment which must be obtained for any computer installation desiring to reproduce sound from numbers. Likewise a suitable output program must be written to transmit the numbers from the computer memory into the converter. The details of this program depend completely upon the particular computer installation and analog-to-digital converter which are used. Consequently, the output program must be written by the computer user and is not included in this group of music programs.

In the music program the sequence of numbers corresponding to musical sounds are generated by simulated instruments built up from combinations of unit generators. Each unit generator is a small block of computer instructions performing a given operation such as that of an oscillator, an adder, or a random noise generator. Figure 2 shows a diagram of a typical simulated instrument. It produces periodic tones in which the wave shape can be controlled, the attack and decay characteristics can be controlled, and frequency can be varied by means of both a periodic and a random vibrato. The instrument specifications in terms of the coding necessary to specify ten of these instruments is indicated at the bottom of the figure. The first line gives the name of the instrument, WAIL; the next six lines specify the unit generators in the instrument. Each unit generator is specified by its type, for example, ØSCIL, standing for oscillator. The interconnectivity of the unit generators is specified by giving the origin of the inputs to each unit generator. The first oscillator U1, for example, has inputs P4 and C3 which are supplied by the composer on a note card at the beginning of each note to be played by the instrument. Oscillator U5, by contrast, has as inputs the output of oscillator U1 and adder U4. The computer description of the unit generators is completed by specifying any other parameters which they may require. For example, the oscillators refer to functions F1 through F20 which are stored in the computer memory and which determine the wave shape of the oscillation.

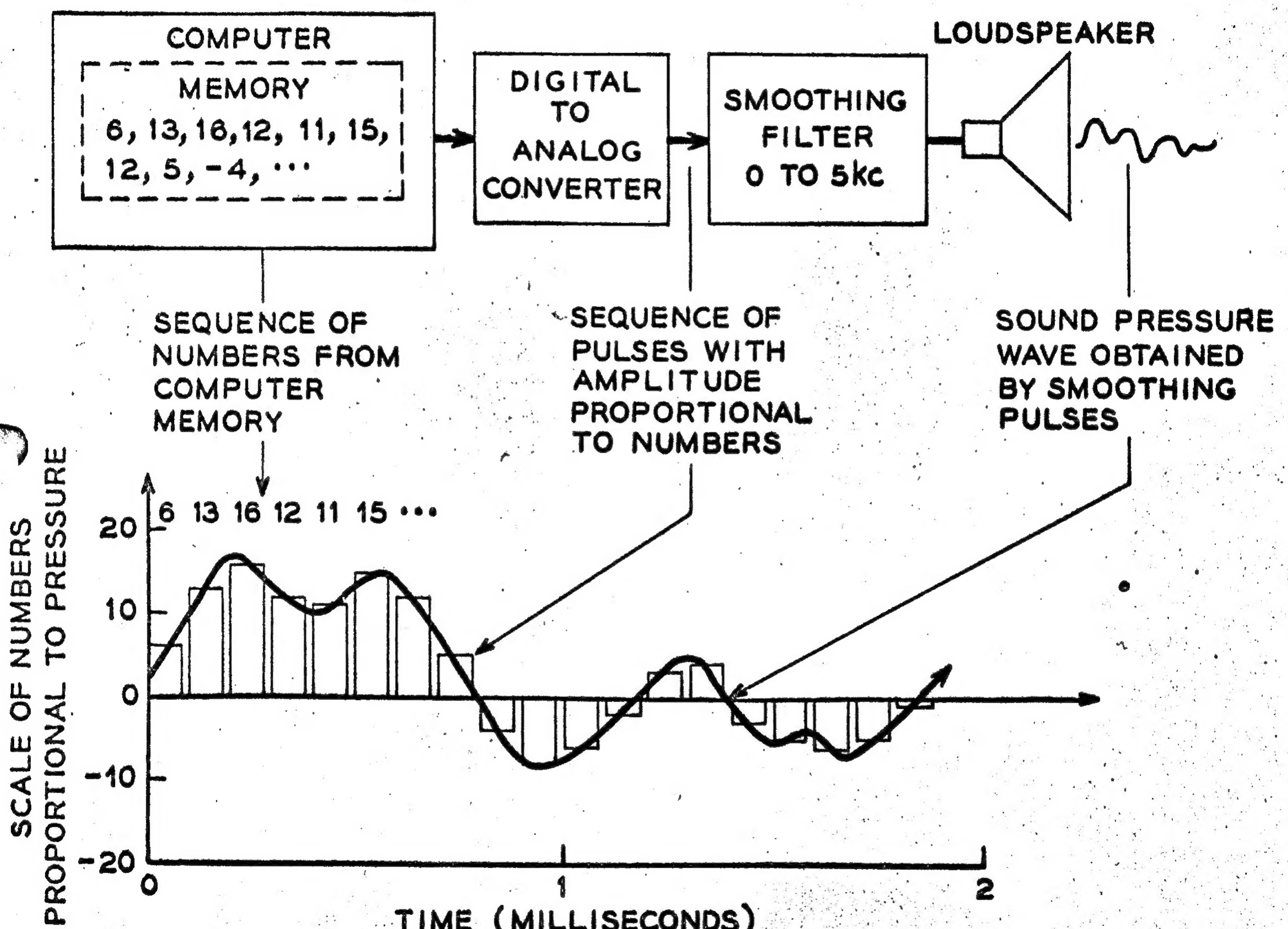
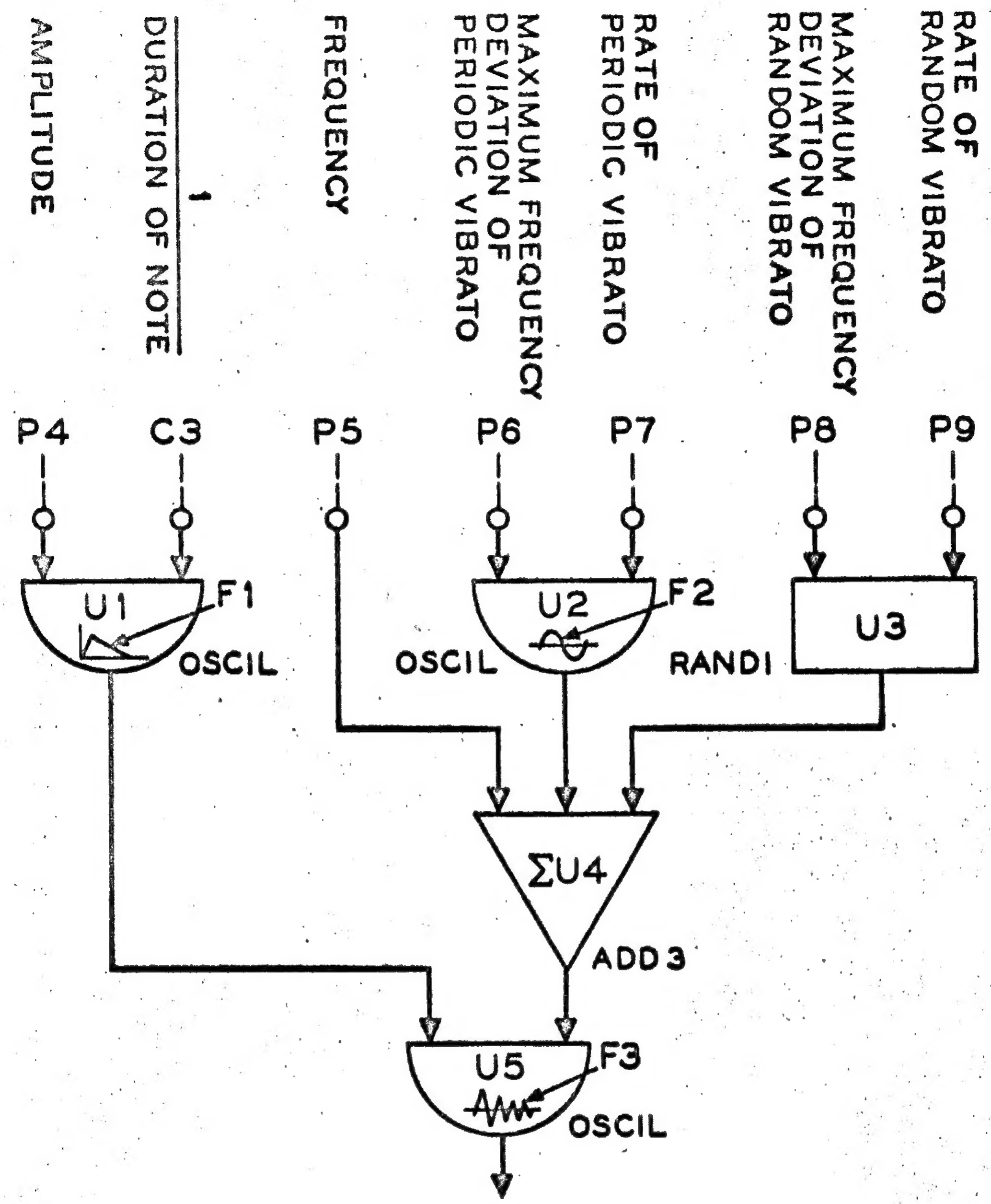


Fig 1
Mathews

BLOCK DIAGRAM AND COMPUTER CODE FOR 10 WAILS



WAIL	INSTR	
	ØSCIL	P4, C3, F1
	ØSCIL	P6, P7, F2
	RANDI	P8, P9
	ADD3	P5, U2, U3
	ØSCIL	U1, U4, F3
	ØUT	U5
	END	
WAIL	CØUNT	10
	FINE	

Fig 2

NOTE CARDS

OP	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	P11	P12
PLA	2	0	0.75	1	466	0	0	7.0	6			
PLA	3	0.5	0.50	2	116	1.7	6	0	0			

INSTRUMENT NO.
STARTING TIME (BEATS)
DURATION (BEATS)
LOUDNESS (ARBITRARY SCALE)
FREQUENCY (CPS)
PERIODIC VIBRATO AMPLITUDE (CPS)
PERIODIC VIBRATO FREQUENCY (CPS)
RANDOM VIBRATO AMPLITUDE (CPS)
RANDOM VIBRATO BANDWIDTH (CPS)
UNUSED

Fig 3

The computer instructions resulting from the instrument specifications are produced by the computer by executing BE FAP, a compiling program. Tn this compilation there are many self-checking features which detect most types of errors the composer may make. For example, if he lists either too many or too few parameters for a given unit generator or puts down the wrong kind of parameter, say a function instead of an input parameter, the compiler will inform the composer of his error and refuse to proceed. A successful compilation results in an orchestra program.

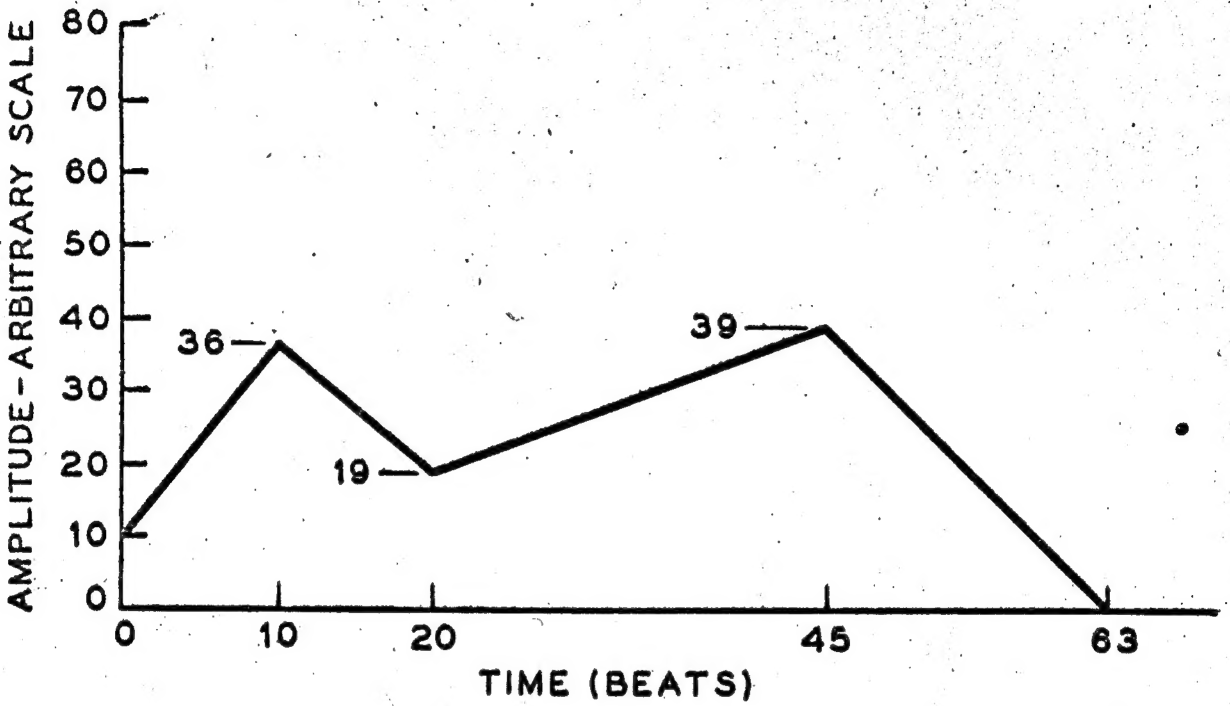
Ten instruments all having the same form are compiled by the single specification WAIL CØUNT 10. The last line FINE indicates that there are no additional instruments in the orchestra.

After the orchestra has been compiled, a score consisting of notes to be played by the instruments is "played" by means of a set of playing programs which operate together with the orchestra program. Notes to be played by the instruments can be generated by the computer using automatic composing features of the playing program or the notes can be punched manually, one note per computer card. In this case each card has room for an operation code and 12 numerical parameters as shown on Figure 3. Here two notes are played on the WAIL instruments #2 and #3. The instrument number is given in P1, the starting time in P2, and the duration in P3. Both starting time and duration are given in beats and the beats are converted into seconds by an arbitrary velocity transformation. Loudness in P4 is given on an arbitrary scale. Note frequency is given in P5 in cycles per second. The amplitudes and rates of the periodic and random vibratos are given in P6 and P9. On this particular instrument, P10 through P12 are not used. In more complicated instruments they may be used.

In addition to specifying notes the computer cards may be used to specify compositional functions. These are functions which change over the course of the composition and are typically used to control parameters such as tempo or loudness. The format for such functions is indicated on Figure 4. They are built up from line segments. On the computer card the time and amplitude of the end point of each line segment is specified. Many very short line segments can be used to describe rapidly changing portions of a function and a few long line segments can be used for the slowly changing portions. Thus an efficient coding is achieved which requires a minimum of writing on the part of the composer.

The operation of the playing program is schematized on Figure 5. The program has been divided into three parts, Pass I, Pass II, and Pass III, to facilitate the use of the computer as a

COMPOSITIONAL FUNCTION

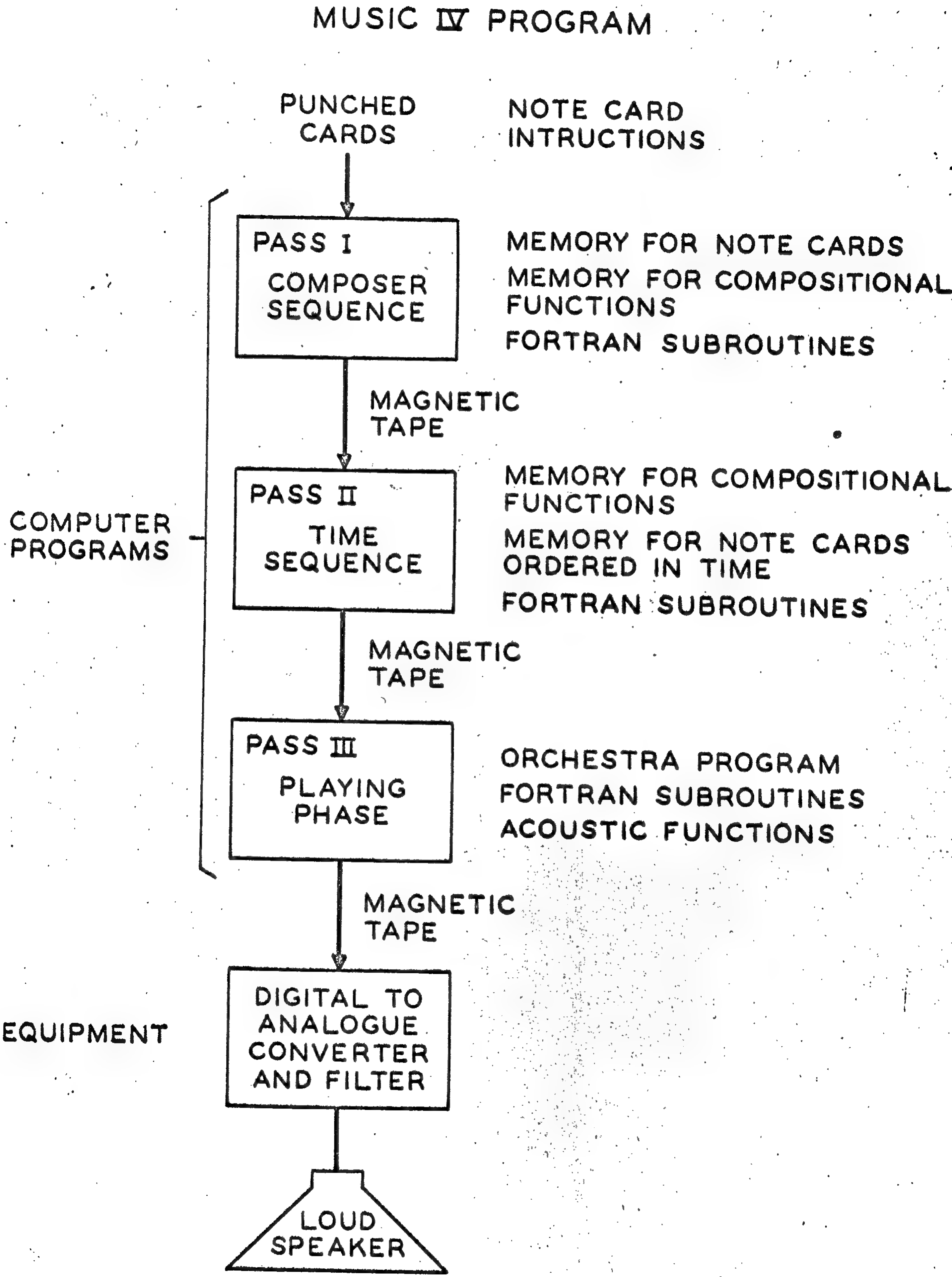


OP	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	P11	P12
110	0	0	0	10	10	36	20	19	45	39	63	0

composing medium. The input to Pass I consists of the cards which the composer prepares. These are note cards such as were shown on Figure 3 plus instruction cards which specify how the note cards will be modified before they are played. Pass I contains a memory for 800 note cards thus allowing sequences of notes to be replayed with or without modification. The same memory also may be used to store compositional functions which are involved in modifying the note cards. The actual modifications are performed by FØRTRAN subroutines. FØRTRAN has proven to be a very satisfactory language in which to write many of the music programs.

In Pass I the cards are processed in the sequence in which the composer has inserted them into the computer. In Pass II the notes are sorted into the time sequence in which they will be played. Consequently, in Pass II it is convenient to apply transformations which are functions of time such as, for example, altering the tempo.

Pass III is the playing phase in which the notes specified directly by the composer or generated by the computer are presented to the orchestra program. The output is shown being directly converted to sound. It is more likely that the numerical samples will be stored on a magnetic tape or disk for subsequent conversion to sound. Also the figure shows communication between the various Passes by means of digital magnetic tapes. If a disk is available it will do these jobs better. Details such as these can and indeed must be altered by the particular user of the program to fit his computer installation.



MUSIC IV PROGRAM
PUNCHED CARDS
NOTE CARD INTRUCTIONS
PASS I
COMPOSER SEQUENCE
MEMORY FOR NOTE CARDS
MEMORY FOR COMPOSITIONAL FUNCTIONS
FORTRAN SUBROUTINES
MAGNETIC TAPE
PASS II
TIME SEQUENCE
MEMORY FOR COMPOSITIONAL FUNCTIONS
MEMORY FOR NOTE CARDS ORDERED IN TIME
FORTRAN SUBROUTINES
COMPUTER PROGRAMS
MAGNETIC TAPE
PASS III
PLAYING PHASE
ORCHESTRA PROGRAM
FORTRAN SUBROUTINES
ACOUSTIC FUNCTIONS
MAGNETIC TAPE
EQUIPMENT
DIGITAL TO ANALOGUE CONVERTER AND FILTER
LOUD SPEAKER

Section 5. Operating procedure.

It is assumed at this point that the following preparations have been made:

1. The absolute BINARY program of BE FAP (File 1) has been used to assemble the CRUNCH deck of BE FAP (File 2) and the BE FAP compiler is available for use (See Section 4);

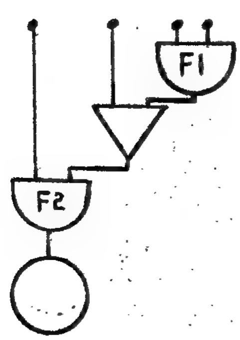
2. Files 3 and 4 have been punched, and the FAP routines DREAD and SØRT have been assembled with BE FAP and the FØRTRAN routines of File 4 have been compiled.

It remains (1) to design, define and compile with BE FAP an orchestra of the composer's choice and (2) to run the three passes of the music program. We shall discuss first the

Preparation of an orchestra

As has been indicated earlier, the orchestra is designed by the composer. The orchestra compiler is a CRUNCH deck of BE FAP coding (given in part 1 File 3 of tape) and the composer writes additional cards defining the instruments in the orchestra of his choice. The CRUNCH deck together with the cards defining the orchestra are then treated as one source deck to be compiled by BE FAP. The resulting object deck, which is to be run in Pass III is thus tailored to the requirements of the composer.

The first step in constructing the orchestra is to design the instruments. An example of an instrument (WAIL) has already been given in Section 2. We shall consider here an instrument of even simpler type: one which emits some specified wave shape at any amplitude and frequency where the vibrato of the note can be controlled. The instrument type should be designed in block diagram form where each block is a unit generator. For example, the diagram of the above instrument is as follows:



unit 1 oscillator

unit 2 adder

unit 3 oscillator

unit 4 output box

Oscillators are unit generators which have three parameters. The first two are inputs, which, when read from left to right on the diagram specify the amplitude and frequency respectively of the wave shape emitted by the oscillator. The function number of this wave shape is designated by the third parameter. Adders put out the sum of the input signals. The output box is required to be the last unit in the design of any instrument. In this instrument a function F1 put out by unit 1 is added to a frequency specification for unit 3 in the adder (unit 2). Therefore, the wave form, F2, emitted by unit 3 is frequency modulated by unit 1. Inputs to unit 1 and the left inputs to units 2 and 3 come from note cards in the score, whereas the right inputs to units 2 and 3 and the input to unit 4 come from units 1, 2, and 3 respectively.

After an instrument type has been designed it remains to define the type for the computer. It is necessary first to name the instrument. Any name of up to six alphanumeric characters may be chosen. The first card in the definition contains this name in the location field (cols. 1-6) and INSTR in the operation field (cols. 8-12). The last card is an END card. Between these two is a sequence of cards - one for each of the unit generators in the design. The order of these cards determines the interconnectivity of the unit generators, that is, units which serve as inputs to a given unit should be listed ahead of the given unit. The source of all inputs must be specified for each unit generator in the instrument, and these specifications are listed in the variable field (column 16). The functions, if any, required by the unit are listed immediately after the inputs. Consequently, the definition of the instrument designed above is given by cards 3 through 8 in Table 1.

The list of available unit generators is given in Table 2, together with specifications, diagrams, and descriptions. In writing cards for an instrument definition, the inputs to a unit generator should be listed in the order given by reading from left to right on the diagram. The designation of inputs are of four types:

1. Pn - for those inputs coming directly from the n^{th} field of note cards in the score.

2. Cn - for those inputs coming indirectly from note cards via converting functions where n is the number of the converting function.

3. Un - for those inputs coming from some other unit generator where n is the position of this other unit in the list of units defining the instrument.

4. Kn - for those inputs which are fixed constants where n refers to the n^{th} number in a list of constants. A list of

numerical quantities (in which decimal points are required) may be entered into the orchestra deck by writing the operation code CØNSTS and a list of decimal numbers separated by commas and enclosed in parentheses in the variable field. Such a card must precede all instrument definitions for which there are any unit generators using constant inputs.

Functions associated with units are of two types:

1. Fn - those which are fixed throughout the performance where n is the number of the function.

2. Pn - those which vary in the course of the performance where the function number to be used is given in the n^{th} field of a note card in the score.

A stored function is requested and numbered by a GEN data card read in Pass I. The function is actually computed or generated in Pass III for use in the playing phase of Pass III. Details on generating subroutines, converting functions, and fields for parameters on note cards can be found in Section 6.

To construct an orchestra after having defined the instrument types it is necessary merely to call for the instruments by name in the desired order. A single instrument can be obtained by one card with the name of the type in the operation field. Several instruments of the same type can be called for on a single card by writing the name in the location field, the code CØUNT in the operation field and the desired number of instruments in the variable field.

The score refers to the instruments by number and thus this sequence of cards calling for instrument types must be ordered to correspond to this numbering. If in the score, instrument numbers have been skipped, then positions in the list of instrument names should be marked by cards having DUMMY in their operation fields. There should be one DUMMY card for each instrument omitted. (The CØUNT code is not applicable for DUMMY instruments.)

Either monophonic or two-channel stereophonic digital output signals can be produced by the orchestra. The former requires no specification and the latter may be obtained by preceding all instrument definitions by a card having the operation code STEREØ. If this mode is chosen, then the ØUT units of all instrument types should have a second input which specifies the proportion (0 to 1) of input to channel A (1 minus this proportion is used for channel B). If the orchestra is defined in STEREØ and the second input is omitted from the ØUT boxes, then the channels will be balanced. The digital output signal is multiplexed, that is, the samples of the generated sound are computed for the two channels alternately.

Finally, all definitions and calls for instruments are followed by a card which has FINE in its operation field. Therefore, an orchestra using three instruments as defined above, which are numbered 1, 2, and 5 in the score and which are to play at constant amplitude and produce a balanced stereophonic output would be called for by the cards given in Table 1. To compile the orchestra thus defined these cards written by the composer should be placed directly behind the orchestra CRUNCH deck and the totality should thus be compiled by BE FAP. The resulting object program as a binary card deck may be quite large, and, therefore, tape or disk storage is recommended for the binary card images. (See Section 6 - Orchestra program for advice on the first compilation.)

To run the Music IV program

The Music IV program in its entirety is comprised of three separate programs, referred to as Passes.

Pass I - The deck includes PASSI (main program), DREAD, WRITER, and CØN plus any PLF subroutines of the composer's choice (See Section 6). Input data for this pass is the score, that is, the composition in the form of note cards prepared by the composer, plus additional cards which may call upon subroutines. These cards are processed in the order in which they are read and an intermediate tape, A4, is generated as output of Pass I. (Tape A4 is, of course, specifically associated with only the BTL version, and other systems may use different tapes or disk storage.) This tape contains all the note card images written directly by the composer together with any note card images generated by PLF subroutine. The note card images on A4 can occur in any order relative to their final order of playing.

Pass II - The deck includes PASSII (main program), SØRT, and CØN. The intermediate tape A4 from Pass I is read by PASSII and the note card images are sorted into the sequence in which the notes will be played in Pass III. A velocity scaling to introduce changes in tempo is applied. The adjusted, time-ordered note card parameters are then written onto a second intermediate tape, B2 (again specific to BTL).

Pass III - The deck includes PASSIII (main program), the orchestra deck, the generating routines GEN07, GEN09 and/or others supplied by the composer, convert functions of the composer's choice, any necessary mathematical routines such as square root or trigonometric functions, and an output program for the samples of the generated sound. This routine is not supplied since its operation will depend upon the installation where used. Its requirements are described in Section 6. The intermediate tape B2 of Pass II containing the sorted note card

images is read by the master program and the orchestra program is called upon to generate samples of the acoustic output. These are written on tape A5 by the output routine. (A5 is again specific to BTL.)

Therefore, the sequence of cards that the user should prepare is as follows:

To be compiled by BE FAP	Orchestra CRUNCH deck Instrument definitions (last card - FINE)
To be loaded and run	PASSI (main) DREAD WRITER CØN (PLFn - optional routines supplied by composer)
Data to be read by Pass I	Data - e.g., GEN cards and score
To be loaded and run	PASSII (main) SØRT CØN
To be loaded and run	PASSIII (main) Orchestra program - from compilation above Output program - written by user (CVTon - converting functions - used as needed) (GENon - generating routines - " " ")

As an aid to understanding this sequence of operations we include a printout from a compilation and run at BTL. See Figure 6. Checks in the left-most column denote BE SYS 5 control cards. Much of the printing pertaining to subroutine entry points and origins results from the BE SYS 5 loader. Other outputs are from the Music IV routines and show in Pass I, the note cards for the score and in Pass II, the sorted score. Pass III has no particular printed output except for an indication of sections and a final comment from the output routine. The listing has been annotated and hopefully will be self-explanatory. Other relevant details may be found in the descriptions of the individual programs given in Section 6.

Fig. 6

```
JØB     AA13,.05     JEM     BLDG 2-5                    12 HRS 34 MIN 15 SEC 05/11/64
NØDECK FAP

PAGE 1     PASS 1 CØMMENTS AND ALTER CARD LISTING         12:34:15     05/11/64   JØB AA13

           CRUNCH      / /

PAGE 2                                                   12:34:15     05/11/64   JØB AA13

SUBRØUTINE ENTRY PØINTS
                    26301        INITL
                    26315        GEN
                    26354        INSERT
                    26444        PLAY
                    26516        REMØVE
                    26563        TER

TRANSFER VECTØR
```

00000 236563000060 CVT00
00001 236563000160 CVT01
00002 236563000260 CVT02
00003 236563000360 CVT03
00004 236563000460 CVT04
00005 236563000560 CVT05
00006 236563000660 CVT06
00007 236563000760 CVT07
~~00010 236563001060 CVT08~~
~~00055 236563040560 CVT45~~
00056 236563040660 CVT46
00057 236563040760 CVT47
00060 236563041060 CVT48

provision for
49 converting functions

00061 272545000060 GEN00
00062 272545000160 GEN01
00063 272545000260 GEN02
00064 272545000360 GEN03
00065 272545000460 GEN04
00066 272545000560 GEN05
00067 272545000660 GEN06
00070 272545000760 GEN07
00071 272545001060 GEN08
00072 272545001160 GEN09
00073 272545010060 GEN10
00074 272545010160 GEN11
00075 272545010260 GEN12
00076 272545010360 GEN13
00077 272545010460 GEN14
00100 272545010560 GEN15
00101 272545010660 GEN16
00102 272545010760 GEN17
00103 272545011060 GEN18
00104 272545011160 GEN19
00105 272545020060 GEN20

21 generating routines

00106 624746646360 SPØUT
00107 624725452460 SPEND
00110 624746646301 SPØUT1

output routines

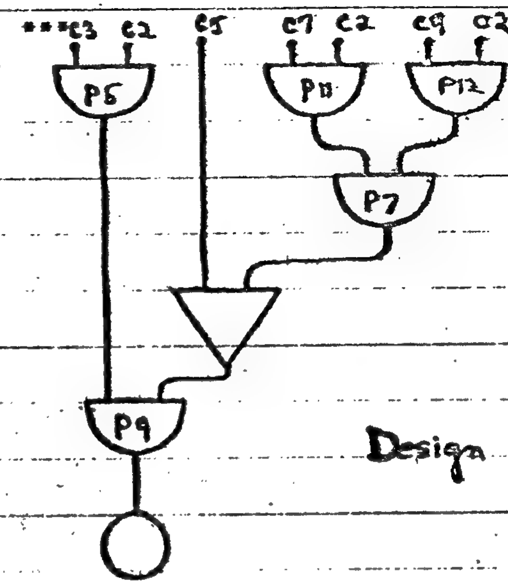
Addresses of Outputs for Unit Generators

```
                         1093                  ***     ØRCHESTRA     ***
                              BELLS INSTR
                                    ØSCIL   C3,C2,P5
                                    ØSCIL   C7,C2,P11
                                    ØSCIL   C9,C2,P12
                                    ØSCIL   U2,U3,P7
                                    ADD2    C5,U4
                                    ØSCIL   U1,U5,P9
                                    ØUT     U6
                                    END
                              BELLS CØUNT   2
                         .06                INSTRUMENT 1
 26602  0 00000 0 26653  .06        PZE ..012+1
 26603  0 00000 0 26703  .06        PZE ..014+1
 26604  0 00000 0 26733  .06        PZE ..016+1
 26605  0 00000 0 26763  .06        PZE ..018+1
 26606  0 00000 0 27011  .06        PZE ..024+1
 26607  0 00000 0 27026  .06        PZE ..028+1
 26610  0 00000 0 27054  .07        PZE ..035+1
                         .07                INSTRUMENT 2
 26611  0 00000 0 27071  .07        PZE ..043+1
 26612  0 00000 0 27121  .07        PZE ..045+1
 26613  0 00000 0 27151  .07        PZE ..047+1
 26614  0 00000 0 27201  .07        PZE ..049+1
 26615  0 00000 0 27227  .07        PZE ..055+1
 26616  0 00000 0 27244  .07        PZE ..059+1
 26617  0 00000 0 27272  .08        PZE ..066+1
 27441                              FINE

LITERALS

  27442 IS THE FIRST LØCATIØN NØT USED BY THIS PRØGRAM

 NØ ERRØR IN ABØVE ASSEMBLY
```



Design of Instrument "Bells"

Fig.6.-3

```
✓ SCR     MYTAPE  A4
✓ SCR     MYTAPE  B2
✓ ØUTPUT  MYTAPE  A5
✓         LØDENS  A5
✓         LØAD
✓         TRA
  PLF1    NØT IN DECK     X
  PLF2    NØT IN DECK     X
  PLF3    NØT IN DECK     X
  PLF4    NØT IN DECK     X
  PLF5    NØT IN DECK     X
  PLF6    NØT IN DECK     X
  PLF7    NØT IN DECK     X
  PLF8    NØT IN DECK     X
  PLF9    NØT IN DECK     X
  PLF10   NØT IN DECK     X
  PLF11   NØT IN DECK     X
  PLF12   NØT IN DECK     X
  PLF13   NØT IN DECK     X
  PLF14   NØT IN DECK     X
  PLF15   NØT IN DECK     X

SUBRØUTINE ENTRY LØCATIØNS       X
  000000  00126   DREAD   00640  WRITER  01325  CØN    01441  (STB)   77026  (WLR)  77031  X
  (SPH)   77014   (FIL)   77024  ENDJØB  77000  (RWT)  77023  SYSTEM  77001               X
                                                                                          X
SUBRØUTINE ØRIGINS       (LØBRK 01665,UPBRK 34642)                                        X
  000000  00100                                                                           X
  DREAD   00640                                                                           X
  WRITER  01314
  CØN     01435
                                                                                          X

PRØTECTED BUFFERS HAVE BEEN SET UP FRØM    10000  TØ
   GEN 09    1   .999    1    0    1   512    0
   GEN 07    2    0     50  .999  412  .999   50      0
   GEN 07    3    0     20  .999  216  .999   40  -.999   216  -.999   0
   ETC                  20    0    0     0     0      0     0      0   0
   GEN 07    4    0    256  .999  256    0     0      0     0      0   0
   GEN 07    5  .999   512  .999    0    0     0      0     0      4   5
         1  .00   .20   3    2   330   1   .01   3   -8   4   5
         1  .25   .20   1    2   330   1   .01   3   -8   4   5
         1  .50   .40   1    2   330   1   .01   3   -8   4   5
         1 1.00   .20   3    2   330   1   .01   3   -8   4   5
         1 1.25   .20   1    2   330   1   .01   3   -8   4   5
         1 1.50   .40   1    2   330   1   .01   3   -8   4   5
         1 2.00   .20   3    2   330   1   .01   3   -8   4   5
         1 2.25   .20   1    2   392   1   .01   3   -8   4   5
         1 2.50   .30   1    2   262   1   .01   3   -8   4   5
         1 2.87   .10   1    2   294   1   .01   3   -8   4   5
         1 3.00   .90   2    2   330   1   .01   3   -8   4   5
         2  0     .20   3    2   262   1   .011  3   -9   4   5
         2  .25   .20   1    2   196   1   .011  3   -9   4   5
         2  .50   .40   1    2   262   1   .011  3   -9   4   5
         2 1.00   .20   3    2   262   1   .011  3   -9   4   5
         2 1.25   .20   1    2   196   1   .011  3   -9   4   5
         2 1.50   .40   1    2   262   1   .011  3   -9   4   5
         2 2.00   .20   3    2   262   1   .011  3   -9   4   5
         2 2.25   .20   1    2   330   1   .011  3   -9   4   5
         2 2.50   .30   1    2   196   1   .011  3   -9   4   5
         2 2.87   .10   1    2   247   1   .011  3   -9   4   5
         2 3.00   .90   2    2   262   1   .011  3   -9   4   5
      TER
```

tape specifications

PASS I main routine

Pass I

Data for PASS I

Note: Checks in leftmost column indicate BE SYSTEM control cards. Crosses on right indicate lines of print produced by BE SYSTEM. All other lines result from MUSIC II.

Note: Encircled numbers were not punched on data cards but result from carry feature of DREAD.

Fig 6-4

Pass II

```
LØAD
TRA
CØN    NØT IN DECK
```

← Note: CØN is usually present in Pass II

```
PLS1   NØT IN DECK
PLS2   NØT IN DECK
PLS3   NØT IN DECK
PLS4   NØT IN DECK
PLS5   NØT IN DECK
PLS6   NØT IN DECK
PLS7   NØT IN DECK
PLS8   NØT IN DECK
PLS9   NØT IN DECK
PLS10  NØT IN DECK
PLS11  NØT IN DECK
PLS12  NØT IN DECK
PLS13  NØT IN DECK
PLS14  NØT IN DECK
PLS15  NØT IN DECK

SUBRØUTINE ENTRY LØCATIØNS
 000000  00133   SØRT    05425   (RWT)   77023   (SPH)   77014   (FIL)   77024   (TSB)   77030
 (RLR)   77027   ENDJØB  77000   (STB)   77026   (WLR)   77031   SYSTEM  77001   EXP(3   05502
 EXPA    05502   PRNTXX  05560   XXMATH  05556   LØNE    05616   LØGE    05646   LØG10   05631
 LØG     05646   EXP     05777

SUBRØUTINE ØRIGINS      (LØBRK 06117,UPBRK 15466)
 000000  00100
```

← Pass II main routine

```
 SØRT    05425
 EXPA    05477
 XXMATH  05556
 LØG     05615
 EXP     05776
```

provided by SYSTEM for exponentials appearing in Pass II

```
IØ BUFFERS HAVE BEEN SET UP FRØM   06127  TØ   15216

SECTIØN NØ    1
```

GEN	9.000	1.000	0.999	1.000	0.	1.000	512.000	0.	-0.	-0.	-0.	-0.
GEN	7.000	2.000	0.	50.000	0.999	412.000	0.999	50.000	0.	-0.	-0.	-0.
GEN	7.000	3.000	0.	20.000	0.999	216.000	0.999	40.000	-0.999	216.000	-0.999	0.
ETC	-0.	-0.	-0.	20.000	0.	0.	0.	0.	0.	0.	0.	0.
GEN	7.000	4.000	0.	256.000	0.999	256.000	0.	0.	0.	0.	0.	0.
GEN	7.000	5.000	0.999	512.000	0.999	0.	0.	0.	0.	0.	0.	0.
NØT	1.000	0.	0.200	1000.000	2.000	330.000	1.000	0.010	3.000	-8.000	4.000	5.000
NØT	2.000	0.	0.200	1000.000	2.000	262.000	1.000	0.011	3.000	-9.000	4.000	5.000
NØT	1.000	0.250	0.200	333.333	2.000	330.000	1.000	0.010	3.000	-8.000	4.000	5.000
NØT	2.000	0.250	0.200	333.333	2.000	196.000	1.000	0.011	3.000	-9.000	4.000	5.000
NØT	1.000	0.500	0.400	333.333	2.000	330.000	1.000	0.010	3.000	-8.000	4.000	5.000
NØT	2.000	0.500	0.400	333.333	2.000	262.000	1.000	0.011	3.000	-9.000	4.000	5.000
NØT	1.000	1.000	0.200	1000.000	2.000	330.000	1.000	0.010	3.000	-8.000	4.000	5.000
NØT	2.000	1.000	0.200	1000.000	2.000	262.000	1.000	0.011	3.000	-9.000	4.000	5.000
NØT	1.000	1.250	0.200	333.333	2.000	330.000	1.000	0.010	3.000	-8.000	4.000	5.000
NØT	2.000	1.250	0.200	333.333	2.000	196.000	1.000	0.011	3.000	-9.000	4.000	5.000
NØT	1.000	1.500	0.400	333.333	2.000	330.000	1.000	0.010	3.000	-8.000	4.000	5.000
NØT	2.000	1.500	0.400	333.333	2.000	262.000	1.000	0.011	3.000	-9.000	4.000	5.000
NØT	1.000	2.000	0.200	1000.000	2.000	330.000	1.000	0.010	3.000	-8.000	4.000	5.000
NØT	2.000	2.000	0.200	1000.000	2.000	262.000	1.000	0.011	3.000	-9.000	4.000	5.000
NØT	1.000	2.250	0.200	333.333	2.000	392.000	1.000	0.010	3.000	-8.000	4.000	5.000
NØT	2.000	2.250	0.200	333.333	2.000	330.000	1.000	0.011	3.000	-9.000	4.000	5.000
NØT	1.000	2.500	0.300	333.333	2.000	262.000	1.000	0.010	3.000	-8.000	4.000	5.000
NØT	2.000	2.500	0.300	333.333	2.000	196.000	1.000	0.011	3.000	-9.000	4.000	5.000
NØT	1.000	2.870	0.100	333.333	2.000	294.000	1.000	0.010	3.000	-8.000	4.000	5.000
NØT	2.000	2.870	0.100	333.333	2.000	247.000	1.000	0.011	3.000	-9.000	4.000	5.000
NØT	1.000	3.000	0.900	666.667	2.000	330.000	1.000	0.010	3.000	-8.000	4.000	5.000
NØT	2.000	3.000	0.900	666.667	2.000	262.000	1.000	0.011	3.000	-9.000	4.000	5.000
TER												

parameters for generating routines

Sorted score

Fig. 6-5

to load in compiled orchestra

Pass III

```
        LØAD     BATCH
        LØAD
        TRA
  CVT00  NØT IN DECK
  CVT01  NØT IN DECK
  CVT04  NØT IN DECK
  CVT06  NØT IN DECK
  CVT08  NØT IN DECK
  CVT10  NØT IN DECK
  CVT11  NØT IN DECK
  CVT12  NØT IN DECK
  CVT13  NØT IN DECK
  CVT14  NØT IN DECK
  CVT15  NØT IN DECK
  CVT46  NØT IN DECK
  CVT47  NØT IN DECK
  CVT48  NØT IN DECK
  GEN00  NØT IN DECK
  GEN01  NØT IN DECK
  GEN02  NØT IN DECK
  GEN17  NØT IN DECK
  GEN18  NØT IN DECK
  GEN19  NØT IN DECK
  GEN20  NØT IN DECK

SUBRØUTINE ENTRY LØCATIØNS
  TER     26663   REMØVE  26616   PLAY    26544   INSERT  26454   GEN     26415   INITL   26401
  000000  27556   GEN07   30111   GEN09   30323   CVT02   30576   CVT03   30632   CVT05   30656
  CVT07   30704   CVT09   30732   (TSB)   77030   (RLR)   77027   (SPH)   77014   (FIL)   77024
  SYSTEM  77001   ENDJØB  77000   CØSD    30755   SIND    30757   SPEND   31327   SPØUT   31233
  SPØUT1  31217   HMPTY   31133   SPIN    31007   SPIN1   30773   CØS     32172   SIN     32174
  PRNTXX  32334   XXMATH  32332   XMATH1  32371

SUBRØUTINE ØRIGINS     (LØBRK 32434,UPBRK 57034)
  INITL   00100
  000000  27542
  GEN07   30105
  GEN09   30316
  CVT02   30572
  CVT03   30626
  CVT05   30652
  CVT07   30700
  CVT09   30726
  SIND    30754
  SPIN1   30773
  SIN     32167
  XXMATH  32332
  XMATH1  32371

PRØTECTED BUFFERS HAVE BEEN SET UP FRØM   40000  TØ
  SEC
  TER
END ØUTPUT FILE        0 SAMPLES ØUT ØF RANGE

DATE  TIMES: ØN      ØFF      ELAPSED  SETUP  0  CHARGED  1401    SYSTEM  5  FØRTRAN  0  FAP  1   RUN  3   DUMPS   PIT  PØT  CØDE
05/11 AA13   12.5708 12.6000  0.0292   0.        0.0292   0.0172  0.0136     0.          0.0105  0.0050   0.0001  6431 3482 44BC1
 PRINTED ØN 1401 SYSTEM 3
```

000000 27542: PASS III main routine

SIND 30754, SPIN1 30773: origin for output routines SPØUT1, SPØUT, SPEND

SIN, XXMATH, XMATH1: routines supplied by SYSTEM for SIN function used in GEN09

FAP 0.0105: hrs. — Compilation time (orchestra)

RUN 0.0050: hrs. — run time

Table 1

Example of an Orchestra Definition

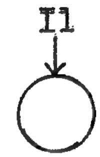
Cols / Cards	1 Location field	8 Operation field	16 Variable field
1		STEREØ	
2		CØNSTS	(5.)
3	WØBBLE	INSTR	
4		ØSCIL	P6, P7, F1
5		ADD2	C4, U1
6		ØSCIL	K1, U2, F2
7		ØUT	U3
8		END	
9	WØBBLE	CØUNT	2
10		DUMMY	
11		DUMMY	
12		WØBBLE	
13		FINE	

Table 2

Unit Generator Specifications

Note: The inputs to the unit generators are described in general by In where n indicates the number of the input counting from left to right on the diagram. Functions, if any, follow the inputs and are denoted by G. The subscript i used in the description of the units denotes sample number.

1. ØUT I1



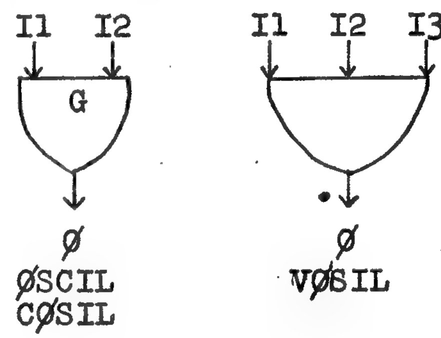
This unit generator must be the last unit in all instruments. It adds the sample specified by I1 to the final acoustical output.

In the STEREØ mode, ØUT may have two inputs ØUT I1,I2 in which I2 (between 0 and 1) specifies the proportion of the instrument's output going to channel A and 1-I2 specifying the proportion of the output going to channel B. (If I2 is omitted, half the output goes to both channels.)

2. ØSCIL I1,I2,G

3. CØSIL I1,I2,G

4. VØSCIL I1,I2,I3



These generate functions and oscillations according to the equations

$$\left.\begin{matrix}\text{ØSCIL}\\ \text{CØSIL}\end{matrix}\right\} \quad \emptyset_i = I1_i * G([S_i] \bmod 512)$$

$$S_{i+1} = S_i + I2_i$$

In ØSCIL, S_0 is set to zero at the beginning of each note. In CØSIL it is not.

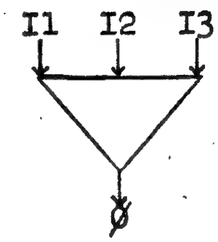
When these units are used as oscillators, I1 specifies the amplitude of the oscillation and 10000xI2/512 is the frequency.

For VØSCIL

$$\emptyset_1 = I1_1 * I3\ ([S_1] \bmod 512),$$

thus the function number is considered as an input and can be changed during the course of the note. If $I3_1$ is not an integer, it is truncated to the next smallest integer.

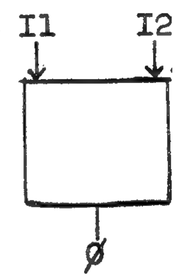
5. ADD2 I1,I2
6. ADD3 I1,I2,I3
7. ADD4 I1,I2,I3,I4



These units add. Thus for example in ADD3

$$\emptyset_1 = I1_1 + I2_1 + I3_1$$

8. RANDI I1,I2



This generator produces a function

$$\emptyset_1 = I1_1 \times \text{Random Function}(I2_1)$$

where Random Function($I2_1$) is a low pass random function having a bandwidth of approximately 5000xI2/512 cps and an amplitude ranging from -1 to +1. The function is formed by connecting independent random numbers with line segments. The number of samples on each line segment is 512/I2.

9. RANDH I1,I2

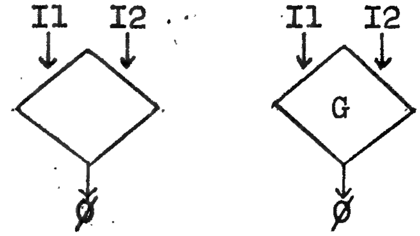
This generator produces a function

$$\emptyset_1 = I1_1 \times \text{Random Number}(I2_1)$$

where Random Number($I2_1$) is a succession of independent random numbers that change every 512/I2 sample. In other words this generator holds each random number for 512/I2 samples.

10. MULT I1,I2

11. VFMULT I1,I2,G

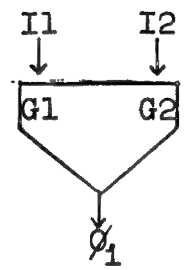


These multipliers perform according to the equations

MULT $\O_i = I1_i * I2_i$

VFMULT $\O_i = I1_i * G(I2_i)$

12. RESØN I1,I2,G1,G2



This generator produces oscillations which are the product of function scanned at a fixed rate times a function scanned at a variable rate according to the equation

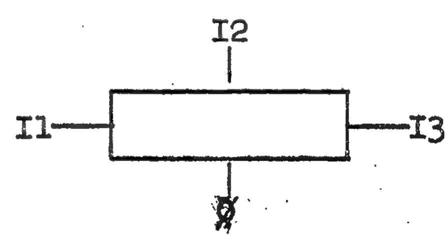
$$\O_i = I1_i * G1([S_i] \bmod 512) * G2([T_i] \bmod 512)$$

$$S_{i+1} = S_i + 1$$

$$T_{i+1} = T_i + I2_i$$

The function G1 can be used to introduce fixed formant frequencies. The function G2 is usually of a type $1/2(1 - \cos 2\pi\, x/512)$ which goes to zero at x=1 and x=512 and thus smoothes any discontinuities in F1.

13. FILTER I1,I2,I3



This unit simulates a single pole pair bandpass filter with the difference equation

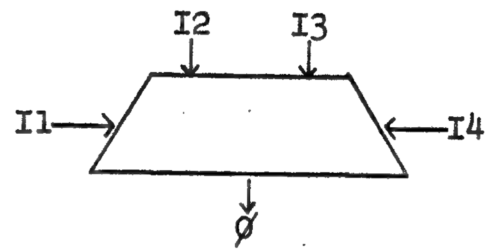
$$\O_i = I2_i + I1_i \O_{i-1} - I3_i \O_{i-2}$$

Normally $I1 = 2e^{-BT}\cos WT$

$$I3 = e^{-2BT}$$

where 2B is the bandwidth of the filter in radians/sec, W is the center frequency in radians/sec, and T is the sampling interval (1/10000 sec).

14. LINEN I1,I2,I3,I4



This generator produces linear attack and decay envelopes for notes where the attack and decay times are independent of the note duration.

I1 rise time in samples

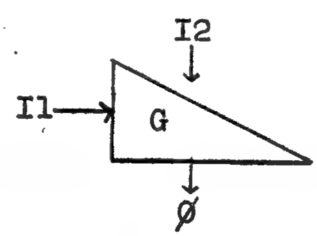
I2 amplitude of the steady state output

I3 duration of entire note in seconds

I4 decay time in samples

10000I3-I4 must be less than 32768. The envelope will not reach full height if $10000I3 < I1+I4$. If $10000I3 < I1$, then the decay time will be zero.

15. EXPEN I1,I2,G



This unit is most frequently used to generate exponential envelopes according to the equation

$$\text{Ø}_i = I1_i * G([S_i]_{TRUNCATED})$$

$$S_{i+1} = S_i + I2_i$$

S_o is set to zero at the beginning of each note where G is an exponential function. $[S_i]_{TRUNCATED}$ holds at 512 for all $S_i > 512$.

EXPEN can also be used for other functions which are to terminate rather than repeat.

Section 6. Music IV routines.

Index to Routine Write-ups

6.1 PASSI (main program)

Description - This program reads the score cards which the composer prepares and writes the information derived from these score cards on an intermediate tape, A4, which is subsequently processed in Pass II.

Subroutines which must be provided

DREAD - described in Section 6

WRITER - described in Section 6

ENDJØB - described in Section 6

SYSTEM - described in Section 6

Use of Common Storage

```
CØMMØN T,P,G

DIMENSIØN T(200),P(20),G(12,800)
```

Details of Operation

The format for the score cards consists of an ØP code plus 12 fields for parameters, P1 through P12, arranged in the card columns as shown:

ØP	P1	P2	P3		P12
1-3	4-6	7-12	13-18	etc.	67-72

On cards which directly play notes, P1 is the instrument number; P2 is the starting time of the note measured from the beginning of each section; P3 is the duration of the note. Both P2 and P3 are written in arbitrary time units called beats and the equivalence between beats and seconds is made in Pass II.

Two types of memory are provided in Pass I. The first consists of memory for 800 note card images G(12,800). Entries are made in this memory by putting a number N (1 through 800) in the ØP code field of the note card. The numbers P1 through P12 then are stored in G(1,N) through G(12,N) respectively. A second memory consists of 200 numbers T(200). The parameters T(101) through T(200) are initialized to a value of 1.0 at the beginning of the composition. The parameters T(1) through T(100) are initialized to a value of 0 at the beginning of each section. Any T parameter may be set by means of a card,

ØP	P1	P2
SPF	n	m

where n is the parameter (1 through 200) and m is the value to which it will be set.

A storage space P(20) is provided in common for communication with subroutines. The note card images are read into this space by DREAD. The parameters P1 through P12 go into locations P(1) through P(12). A numerical equivalent of the ØP code goes into P(13). The equivalence table is as follows:

ØP Code	Numerical Equivalence	Meaning
bbb*	1	Play a Note in Pass III
GEN	4	Generate Function in Pass III
SPF	5	Set Parameter in Pass I
TER	6	Terminate Composition
PLF	8	Call Subroutine in Pass I
SEC	9	Terminate Section
SPS	10	Set Parameter in Pass II
ETC	11	Continuation Card
PLS	12	Call Subroutine in Pass II
n	13	Store a Note Card in Pass I or Pass II. n is integer between 1 and 999.

If P(13) = 13, then the number N of the G(12,N) storage array is stored in P(14).

The P array is also used to write data on intermediate tape A4 to communicate with Pass II. For details see the subroutine WRITER. There is a general correspondence between the parameters written on A4 and the parameters written by DREAD into P. P(1) - P(3) are instrument number, starting beat, and duration. P(13) is a numerical ØP code. For GEN cards, P(2) is not the starting time, but rather the starting time in beats is put in P(14).

*bbb stands for a blank ØP code field.

WRITER also takes care of parameters T(1) through T(32). T(1) through T(30) contain the termination times of the note with the latest termination that has been played on instruments 1 through 30 respectively. T(31) contains the termination time of the latest terminating note already played in the section. T(32) contains a count of the number of notes already played. A maximum of a thousand notes may be played in a section and WRITER will terminate the composition with a nasty comment if this number is exceeded.

The operation of the various ØP codes will next be described even though they do not all operate in Pass I.

bbb - Play a note in Pass III

bbb P1 P2 P3 P4...P12

This card causes a note to be played on an instrument P1 starting at time P2 and lasting for duration P3. A note card image is generated on intermediate tapes A4 and B2.

GEN - Generate function in Pass III

GEN P1 P2 P3...P12

This causes the generation of a function in Pass III with subroutine P1. The function number is P2. The subroutine may use parameters P3 through P12. If additional parameters are required they may be written on ETC cards following the GEN card. The function is generated at the time in Pass III equal to the value of T(31) in Pass I when the GEN card was read.

SPF - Set parameter in Pass I

SPF P1 P2

This card causes T(P1) to be set to the value P2 in Pass I.

TER - Terminate composition

This card terminates the composition. In Pass I it terminates the first pass and transfers control by CALL SYSTEM; in Pass II it terminates the second pass and transfers control by CALL SYSTEM; in Pass III it terminates the acoustic output tape and transfers control by CALL SYSTEM. It is also equivalent to an SEC card for the last section in the composition.

PLF - Call subroutine in Pass I

PLF P1 P2...P12

This card calls upon subroutine PLFP1 in Pass I where P1 is an integer, 1 to 15. At the subroutine execution time the parameters P1 through P12 are in the P(20) array. The PLF subroutines which are called upon must, of course, be supplied by the programmer.

SEC - Terminate section

This card terminates a section, resets the timeclock of the compiler and reinitializes T(1) through T(100) to 0. Thus, time which is written in P2 of the note cards is always measured from the beginning of the current section. No more than a thousand notes may be played in a section. This limit is imposed by the sorting in Pass II which is done a section at a time.

SPS - Set parameter in Pass II

SPS P1 P2

This card sets parameter T(P1) to the value P2 in Pass II.

ETC - Continuation card

ETC P1...P12

This card is used for continuing the number of parameters beyond 12. The parameters on the ETC card are always carried along with the previous card. In Pass III the P array is increased to P(480) and the parameters are put in successive blocks of 12 locations in this augmented array. Thus, the 12 parameters on the first ETC card would go into P(13) through P(24), the next ETC card into P(25) through P(36), etc.

PLS - Call subroutine in Pass II

PLS P1 P2...P12

This card calls upon subroutine PLSP1 in Pass II where P1 is an integer, 1 to 15. The quantities P2 through P12 will not be in the P(20) array at the time the subroutine is called but they can be referred to by the subroutine. For details read the Pass II FØRTRAN program. The subroutine is executed at the end of the section after the note cards have been sorted into chronological order.

n - Store a note card in Pass I or Pass II

```
n   P1...P12
where n is integer, 1 to 999
```

If n is equal to or less than 800, this card results in the storage of the parameters P1 through P12 in the note card storage space G(1,n) through G(12,n). If n is 801 through 999, this card results in the storage of parameters in the Pass II G array in locations G(12,n-800). Thus, provision has been made for 800 note card images to be stored in Pass I and 199 in Pass II. The storage of a note card image does not directly result in playing a note in Pass III. In order to produce a note one of the PLF subroutines must generate a note card image on tape A4 or B2.

6.10 CVT n Function - Note this is a FØRTRAN function subprogram.

Description - The CVT n functions may be called upon by the orchestra program at the beginning of each note. They convert the parameters from the score card into an input parameter for some unit generator. Function CVT n is called by a statement containing Cn in the orchestra definition. All the CVT routines are written by the user of the music program, and we give here only the form of them.

Subroutines which must be provided

Depends on user.

Use of Common Storage

```
COMMØN P

DIMENSIØN P(480)
```

Details of Operation

When CVT is called, the P array contains the parameters P1-P12 from the score card in locations P(1) through P(12). The parameters from any following ETC cards are in P(13)-P(24), P(25)-P(36), etc. The CVT function may use these values in computing its value.

6.11 GEN07 Subroutine

Description - This program generates a stored function out of straight line segments.

Calling Sequence

CALL GEN07 (A)

A is an array of dimension 512 in which the output is stored.

Subroutines which must be provided

NØNE

Use of Common Storage

CØMMØN P

DIMENSIØN P(480)

Details of Operation

GEN07 computes A(I) for I=1,512 as points on the line segment function described by the P array. The P array designation is as follows

P(3) - A_1		
P(4) - N_1	P(16) - N_5	P(28) - N_9
P(5) - A_2	P(17) - A_6	P(29) - A_{10}
P(6) - N_2	P(18) - N_6	etc.
P(7) - A_3	P(19) - A_7	
P(8) - N_3	P(20) - N_7	
P(9) - A_4	P(21) - A_8	
P(10)- N_4	P(22) - N_8	
P(11)- A_5	P(23) - A_9	

where A_1 is the amplitude of point P_1, N_1 is the abscissa distance from point P_1 to point P_2, A_2 is the amplitude of point P_2, N_2 is the abscissa distance between point P_2 and P_3, etc. The line segment function is formed by connecting points P_1 to P_2, P_2 to P_3, etc. with straight lines. GEN07 terminates the function when either $N_i = 0$ or when the sum of the N_i's is greater than 512.

6.12 GEN09 Subroutine

Description - This program generates a stored function as the sum of segments of sinusoids.

Calling Sequence

CALL GEN09 (A)

A is an array of dimension 512 in which the output is stored.

Subroutines which must be provided

GEN09 is written in FØRTRAN and requires a sine function

SINDF(X)

which produces the sine of an argument given in degrees.

Use of Common Storage

CØMMØN P

DIMENSIØN P(480)

Details of Operation

The parameters of the sinusoids are given in the P array in the following order

P(3) - A_1	P(15) - A_3	etc.
P(4) - H_1	P(16) - H_3	
P(5) - P_1	P(17) - P_3	
P(6) - B_1	P(18) - B_3	
P(7) - E_1	P(19) - E_3	
P(8) - A_2	P(20) - etc.	
P(9) - H_2	P(21) -	
P(10)- P_2	P(22) -	
P(11)- B_2	P(23) -	
P(12)- E_2	P(24) -	

The function is computed according to the equations

$$A(I) = M \cdot \sum F_j(I)$$

$$F_j(I) = \begin{cases} A_j \quad \mathrm{SINFD}\left\{ \frac{360}{512}\left[(I-B_j)*H_j+P_j \right] \right\} \\ \text{for } B_j \leq I \leq |E_j| \\ 0 \quad \text{otherwise} \end{cases}$$

The summation is terminated at the first zero value of A_j. If the first E_i is negative on the last card used to specify the function, then the scaling constant M is set equal to one. If this E_i is positive, the scaling constant is set so that the maximum absolute value of the function is .9999.